TMA03

Question 1

a.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **Possible obstacle** |
| **FR1** | Register | OU user can register for the app using the OU user name | Access to the OU user name database might not be available, they may not wish to share this information with another app, or there may be no api available to access the information. |
| **FR2** | Request ride | The user can request a ride share by submitting relevant details to the app | Lack of access to the internet from the mobile device, for any number of reasons, would stop the app connecting to the server and sending the request. |
| **FR3** | Send ride alert | If two users are on the same train, an alert will show indicating the match | The app might not have permissions to send push notifications, meaning that the user may not get immediate notification of the ride availability. |
| **FR4** | Cancel ride request | The user shall be able to cancel any ride shares that they have requested | If the request is not cancelled in good time the ride volunteer may not receive their request before they expect to meet the other person, meaning they wasted time waiting for their ride companion, when the ride isn’t needed. |
| **FR5** | Take picture | The user is able to take a picture with their device and send with their ride request | The device being used may not be compatible with the camera plugin , due to its version of the Android OS being outdated for instance. |

b. The code in the index.html file so far is as follows, with the changes that I have made high-lighted in blue.:

<body>

<div class="app">

<h1>Taxi Sharing</h1>

<div data-role="controlgroup" data-type="horizontal">

<div>

Name

<input id="name" type="text" />

</div>

<div>

Start

<input id="time" type="text" />

</div>

<div>

End

<input id="time2" type="text" />

</div>

<div>

Pickup Address

<input id="addr" type="text" />

</div>

</div>

<div data-role="controlgroup" data-type="horizontal">

<button type "button" id="register">Register</button>

<button type="button" id="volunteer">Volunteer</button>

<button type="request" id="request"> Request</button>

<button type="button" id="cancel">Cancel</button>

<button type="button" id="start">Start</button>

<button type="button" id="stop">Stop</button>

</div>

</div>

</body>

c. The body of the index.html file is now as follows:

<body>

<div class="app">

<h1>Taxi Sharing</h1>

<canvas style="width:100%;height:50%;" id="map\_canvas">

</canvas>

<div data-role="controlgroup" data-type="horizontal">

<div>

Name

<input id="name" type="text" />

</div>

<div>

Start

<input id="time" type="text" />

</div>

<div>

End

<input id="time2" type="text" />

</div>

<div>

Pickup Address

<input id="addr" type="text" />

</div>

</div>

<div data-role="controlgroup" data-type="horizontal">

<button type "button" id="register">Register</button>

<button type="button" id="volunteer" Volunteer</button>

<button type="request" id="request" Request</button>

<button type="button" id="cancel">Cancel</button>

<button type="button" id="start" >Start</button>

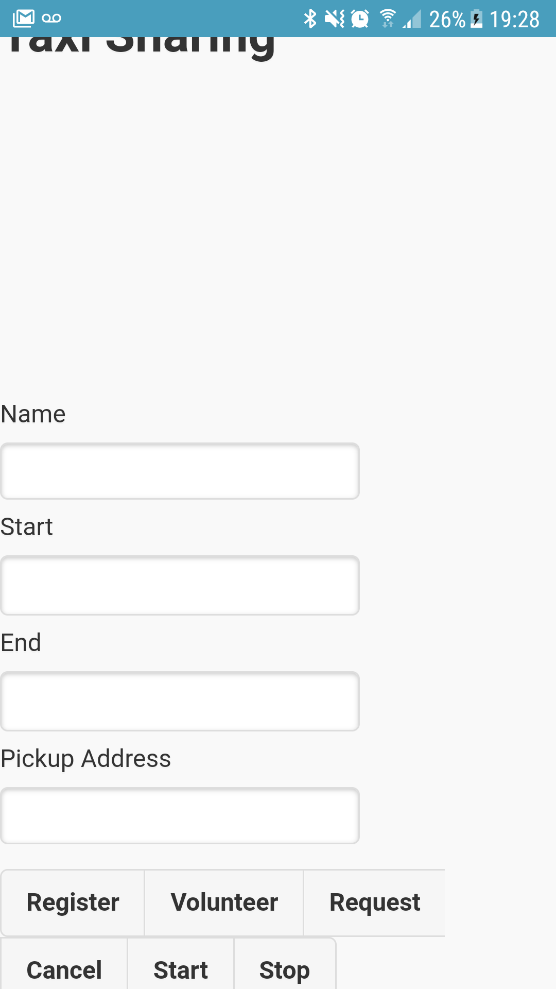
<button type="button" id="stop" Stop</button>

</div>

</div>

</body>

After building the app, the interface showing in the app is:



Question 2

Code for FR 1.2

this.volunteer = function () {

var oucu = get\_name\_value('name', 'user1');

address = get\_name\_value('addr', "Open University")

var start\_time = get\_name\_value('time', format(new Date()));

var end\_time = get\_name\_value('time2', 1);

// compute the date of the next end\_time hours

var d = new Date();

d.setHours(d.getHours() + end\_time);

var ends = format(d);

$.post('http://137.108.93.222/openstack/taxi/orders', {

OUCU: oucu,

start: start\_time,

end: end\_time,

type: 0,

address: address

}, function (data) {

var obj = $.parseJSON(data);

if (obj.status == "fail") {

alert(obj.data[0].reason);

} else {

alert('Taxi has been volunteered');

}

});

};

Code for FR 1.3

this.request = function () {

var oucu = get\_name\_value('name', 'user1');

var address = get\_name\_value('addr', "Open University")

var start\_time = get\_name\_value('time', format(new Date()));

// Post the details of start time, the address and the type to the orders API

$.post('http://137.108.93.222/openstack/taxi/orders', {

OUCU: oucu,

start: start\_time,

type: 1,

address: address

}, function (data) {

var obj = $.parseJSON(data);

if (obj.status == "fail") {

alert(obj.data[0].reason);

} else {

alert('Taxi share has been requested');

}

});

};

Code for FR 2.1

var oucu = get\_name\_value('name', 'user1');

$.get('http://137.108.93.222/openstack/taxi/matches?OUCU=' + oucu,

function (data) {

var obj = $.parseJSON(data);

if (obj.status == "fail") {

alert(obj.data[0].reason);

} else {

if (obj.data.length > 0) {

var parsedData = $.parseJSON(obj.data)

var add = parsedData[0].hire\_address

$.get('http://nominatim.openstreetmap.org/search/' + add + 'format=json&countrycode=gb',

function (result) {

let parsedAdd = $.parseJSON(result)

updateMap(parsedAdd[0]);

});

}

}

});

Code for FR 2.2

if (address != undefined) {

// FR2.2

var meetingLocation = new plugin.google.maps.LatLng(address.lat, address.long);

map.addMarker({

'position': meetingLocation,

'title': "Pickup point"

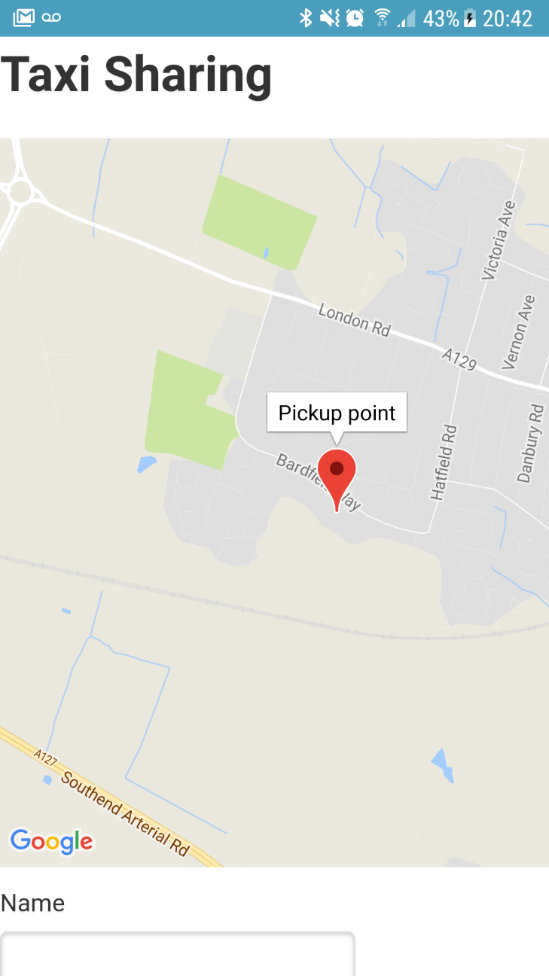
}, function (marker) {

marker.showInfoWindow()

});

}

Screenshot of the app running, showing pickup point on the map



Question 3

1. **FR4 –** When requesting or offering a ride share, the user shall be able to add an image taken with their device’s camera, so that each party in the ride share are able to identify each other.

The following market research was carried out to determine whether the enhancement was useful:

|  |  |
| --- | --- |
| **App** | **Feature Present?** |
| Aya carpooling | Y |
| BlaBla Car | Y |
| Cab with me | Y |

The rationale behind including this additional feature in the app is to improve the user experience, and the ease with which they can achieve the outcome they want from the app. Including an image will ensure that users of the app are able to identify each other easily. Based on the research I carried out, the feature is already widely used in other apps, and therefore I think it would enhance the functionality of my app in this case.

1. My first port of call when selecting an appropriate plugin was the official Cordova plugins store. I felt that this was the most suitable place to find the plugin as they are well documented, meaning implementation will hopefully be straightforward, and it is easy to search and sort possible plugins to find the most suitable. Within the store I selected only plugins that support the Android platform, and then sorted by number of downloads to get an idea of popular features that the community are adding to their apps. One choice of plugin that I considered was that of a spinner, to indicate when the app was busy calling the API’s. I have experienced slow response times myself while testing the app, and thought this would be a good feature for user experience, but wasn’t certain whether this would be considered an enhancement, rather than a requirement. I then settled on using the camera plugin as the ability to add a picture is mentioned in the original description of the app, and it felt like a useful feature for the app to include.
2. The user would use this feature to add an image when they submitted a request or volunteered for a taxi share. The image would enable the users to identify other users that they have matched with in the app. The code I used to implement this feature is:

Index.html

<button type="button" id="photo" onClick="app.taxiShare.takePhoto()">Add Photo</button>

<img style="width:50%;height:20%;" id="photo\_image" />

Index.js

this.takePhoto = function () {

var cameraSuccess = function (imageData) {

var image = document.getElementById('photo\_image');

image.style.display = "block";

image.src = imageData;

};

var cameraError = function () {

console.err("Problem in getting the camera picture");

};

navigator.camera.getPicture(cameraSuccess, cameraError, { quality: 50 });

}

The code in index.html defines two new elements, the first of which is a button that opens the camera on the device. The second is a blank image tag, that is used to display the image once it has been taken with the camera. The code in index.js is triggered when the button is clicked. It accesses the camera in the device using an API exposed by the plugin that has been used. If it successfully takes a picture it is set as the source of the new photo\_image element. If there is a failure while taking the picture an error is logged to the console stating this.

1. The app runs as expected, and I am able to take a picture with the app using the ‘Add Photo’ button. Although this part of the functionality worked ok, I am unable to submit the request including the image to the server, as the API would need to be amended for this to be possible. For this reason, I wasn’t able to fully implement the FR4.

Question 4

a.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **App** | Add Text | Add image | Add video | Plan Upcoming trip | Share to social media |
| Travel Diary  (e250 Apps) | Y | Y | Y | N | N |
| Travel Diary  (Just Rose Travel Diary) | Y | Y | N | N | N |
| VOLO – Your Travel Journal | Y | Y | N | N | Y |
| journi | Y | Y | N | N | Y |

b.

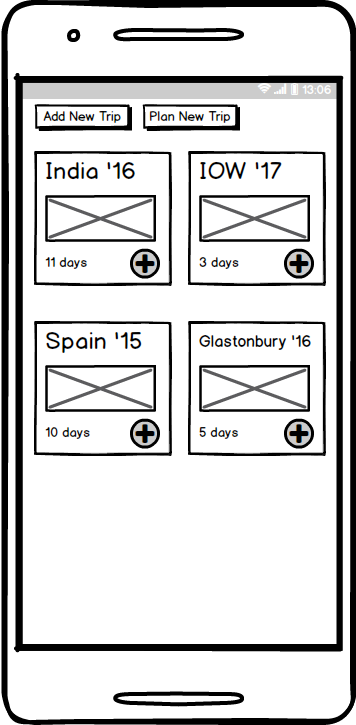
|  |  |  |
| --- | --- | --- |
| **FR1** |  | The app facilitates the addition of an entry to a travel journal. |
|  | FR1.1 | A user can enter a text entry of varying length, to record details of a particular day of travelling/holidaying |
|  | FR 1.2 | A user can upload multiple images to the journal entry to accompany the text |
|  | FR 1.3 | A user can upload short video clips to the journal entry to accompany the text |
| **FR2** |  | The app shall allow a user to share any diary entry via email, or on social media platforms |
| **FR3** |  | The app can facilitate the planning of upcoming trips, using the planner feature |
|  | FR 3.1 | A user can upload email confirmations for travel arrangements, booked excursions etc in a central place |
|  | FR 3.2 | Routes can be planned using a map to plot points |

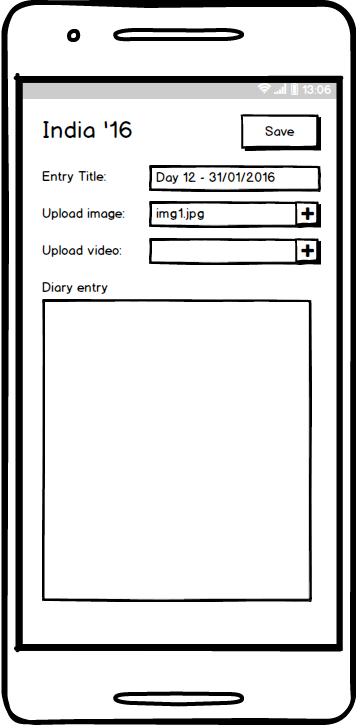
c.

The proposed app would be used by travellers/holiday-makers who want to plan and keep a record of trips they have taken. It would also allow them to plan upcoming trips using a planning feature, that would facilitate the creation of travel itineraries. Users can create trips, and then within the trip, daily entries can be created of the activities they carried out that day. The entry can include text, images and short video clips. Users would then be able to share diary entries via email, or social media platforms.

d.

Some wireframes showing an initial layout are shown below. The first is the main dashboard area where users can see the individual trip journals that they have created, as well as adding a new journal entry for a trip, or creating a new trip. The second screen shows the journal entry screen, and would be accessed by clicking the + icon for the trip that we wanted to add to.





1. Feedback regarding the concept and initial design of the application were very positive, and both interviewees said that they would be very likely to download and use the app if it were created. One interviewer said “It sounds really good, I would definitely download it”, and another said “I have actually been looking for something like this and haven’t been able to find anything good, it would be really useful”. Feedback on the initial wireframes was generally positive, with it seeming clear how to interact with the app. I did however receive suggestions for improvements to the app. The first suggested including the ability to add a map point to a diary entry, and the second that it was missing a way on the dashboard to navigate to future trips that were already in the planning stage. There is a button to create an upcoming trip, but no way to access these trips to view plans. Overall the response seemed very positive to the app.